AMENDMENTS TO THE CLAIMS

The following listing of claims will replace all prior versions, and listings, of claims in the application.

1. (Currently Amended) A method of performing a two-dimensional discrete cosine transform (DCT) using a microprocessor having an instruction set that includes single-instruction multiple-data (SIMD) ~~SIMD~~ floating point instructions, wherein the method comprises:

receiving a block of integer data having C columns and R rows, wherein each of the R rows contains C row data values, wherein the block of integer data is indicative of a portion of an image; and

for each row,

loading the C row data values of the row into registers;

converting the C row data values into floating point form, wherein the registers each hold two floating point row data values; and

performing a plurality of weighted-rotation operations on the values in the registers, wherein the weighted-rotation operations are performed using SIMD floating point instructions.

2. (Currently Amended) The method of claim 1, wherein said converting is accomplished using ~~the pi2fw~~ a packed integer word to floating-point conversion (pi2fw) instruction.

3. (Currently Amended) The method of claim 1, wherein said weighted-rotation operations are accomplished using ~~the pswap, pfmul, and pfpnacc instructions~~ a packed swap doubleword (pswapd) instruction, a packed floating-point multiplication (pfmul) instruction and a packed floating-point negative accumulate (pfpnacc) instruction.

4. (Original) The method of claim 1, further comprising:

for each row,

altering the arrangement of values in the registers;

performing a second plurality of weighted-rotation operations on the values in the registers;

again altering the arrangement of the values in the registers;

performing a third plurality of weighted-rotation operations on the values in the registers;

yet again altering the arrangement of the values in the registers; and

performing a fourth plurality of weighted-rotation operations on the values in the registers to obtain intermediate floating point values.

5. (Original) The method of claim 4, further comprising:

for each row,

storing the intermediate floating point values to an intermediate buffer.

6. (Original) The method of claim 5, further comprising:

for two columns at a time,

loading data from two columns of intermediate data into each of a plurality of registers;

performing a plurality of weighted-rotation operations on the values in the registers, wherein the weighted-rotation operations for two columns are performed in parallel using SIMD floating point instructions.

7. (Currently Amended) The method of claim 6, wherein said weighted-rotation operations for two columns at a time are accomplished using ~~pfmul, pfsub, and pfadd instructions~~ a packed floating-point multiplication (pfmul) instruction, a packed floating-point subtraction (pfsub) instruction and a packed floating-point addition (pfadd) instruction.

8. (Original) The method of claim 6, further comprising:

for two columns at a time,

as each weighted-rotation operation is done, storing weighted-rotation operation results to the intermediate buffer.

9. (Original) The method of claim 8, further comprising:

for two columns at a time,

retrieving weighted-rotation operation results from the intermediate buffer;

performing a second plurality of weighted-rotation operations on the retrieved values;

again storing weighted-rotation operation results to the intermediate buffer as the weighted-rotation operations of the second plurality are done;

again retrieving weighted-rotation operation results from the intermediate buffer;

performing a third plurality of weighted-rotation operations on the retrieved values;

yet again storing weighted-rotation operation results to the intermediate buffer as the weighted-rotation operations of the third plurality are done;

yet again retrieving weighted-rotation operation results from the intermediate buffer;

performing a fourth plurality of weighted-rotation operations on the retrieved values;

converting the weighted-rotation operation results from the fourth plurality to integer results.

10. (Original) The method of claim 9, further comprising:

for two columns at a time, writing the integer results to an output buffer.

11. (Currently Amended) A method of performing a discrete cosine transform (DCT) using a microprocessor having an instruction set that includes single-instruction multiple-data (SIMD) ~~SIMD~~ floating point instructions, wherein the method comprises:

receiving a block of integer data having C columns and R rows; and

for two columns at a time,

loading column data into registers;

converting the column data into floating point form, wherein the registers each hold a floating point column data value from two columns; and

performing a plurality of weighted-rotation operations on the values in the registers, wherein the weighted-rotation operations for two columns are performed in parallel using SIMD floating point instructions.

12. (Currently Amended) The method of claim 11, wherein said weighted-rotation operations are accomplished using ~~pfmul, pfsub, and pfadd instructions~~ a packed floating-point multiplication (pfmul) instruction, a packed floating-point subtraction (pfsub) instruction and a packed floating-point addition (pfadd) instruction.

13. (Original) The method of claim 11, further comprising:

for two columns at a time,

as each weighted-rotation operation is done, storing weighted-rotation operation results to an intermediate buffer.

14. (Original) The method of claim 13, further comprising:

for two columns at a time,

retrieving weighted-rotation operation results from the intermediate buffer;

performing a second plurality of weighted-rotation operations on the retrieved values;

again storing weighted-rotation operation results to the intermediate buffer as the weighted-rotation operations of the second plurality are done;

again retrieving weighted-rotation operation results from the intermediate buffer;

performing a third plurality of weighted-rotation operations on the retrieved values;

yet again storing weighted-rotation operation results to the intermediate buffer as the weighted-rotation operations of the third plurality are done;

yet again retrieving weighted-rotation operation results from the intermediate buffer;

performing a fourth plurality of weighted-rotation operations on the retrieved values;

converting the weighted-rotation operation results from the fourth plurality to integer results.

15. (Original) The method of claim 14, further comprising:

for two columns at a time, writing the integer results to an output buffer.

16. (Currently Amended) A computer system comprising:

a processor having an instruction set that includes single-instruction multiple-data (SIMD) ~~SIMD~~ floating point instructions; and

a memory coupled to the processor, wherein the memory stores software instructions executable by the processor to implement [the method of] a two-dimensional discrete cosine transform method, the method comprising: receiving a block of integer data having C columns and R rows, wherein each of the R rows contains C row data values, wherein the block of integer data is indicative of a portion of an image; and

for each row,

loading the C row data values into registers of the processor;

converting the C row data values into floating point form, wherein the registers each hold two floating point row data values; and

performing a plurality of weighted-rotation operations on the values in the registers, wherein the weighted-rotation operations are performed using SIMD floating point instructions.

17. (Currently Amended) A carrier medium comprising software instructions executable by a microprocessor having an instruction set that includes single-instruction multiple-data (SIMD) ~~SIMD~~ floating point instructions to implement a method of performing a two-dimensional discrete cosine transform (DCT), wherein the method comprises:

receiving a block of integer data having C columns and R rows, wherein each of the R rows contains C row data values, wherein the block of integer data is indicative of a portion of an image; and

for each row,

loading the C row data values into registers;

converting the C row data values into floating point form, wherein the registers each hold two floating point row data values; and

performing a plurality of weighted-rotation operations on the values in the registers, wherein the weighted-rotation operations are performed using SIMD floating point instructions.

18. (New) A computer system comprising:

a processor having an instruction set that includes single-instruction multiple-data (SIMD) floating point instructions; and

a memory coupled to the processor, wherein the memory stores software instructions executable by the processor to implement the method of receiving a block of integer data having C columns and R rows, wherein the block of integer data is indicative of a portion of an image; and

for two columns at a time,

loading column data into registers of the processor;

converting the column data into floating point form, wherein the registers each hold a floating point column data value from two columns; and

performing a plurality of weighted-rotation operations on the values in the registers, wherein the weighted-rotation operations for two columns are performed in parallel using SIMD floating point instructions.

19. (New) A carrier medium comprising software instructions executable by a microprocessor having an instruction set that includes single-instruction multiple-data (SIMD) floating point instructions to implement a method of performing a discrete cosine transform (DCT), wherein the method comprises:

receiving a block of integer data having C columns and R rows; and

for two columns at a time,

loading column data into registers;

converting the column data into floating point form, wherein the registers each hold a floating point column data value from two columns; and

performing a plurality of weighted-rotation operations on the values in the registers, wherein the weighted-rotation operations for two columns are performed in parallel using SIMD floating point instructions.